Neural networks

Training neural networks - activation function derivative

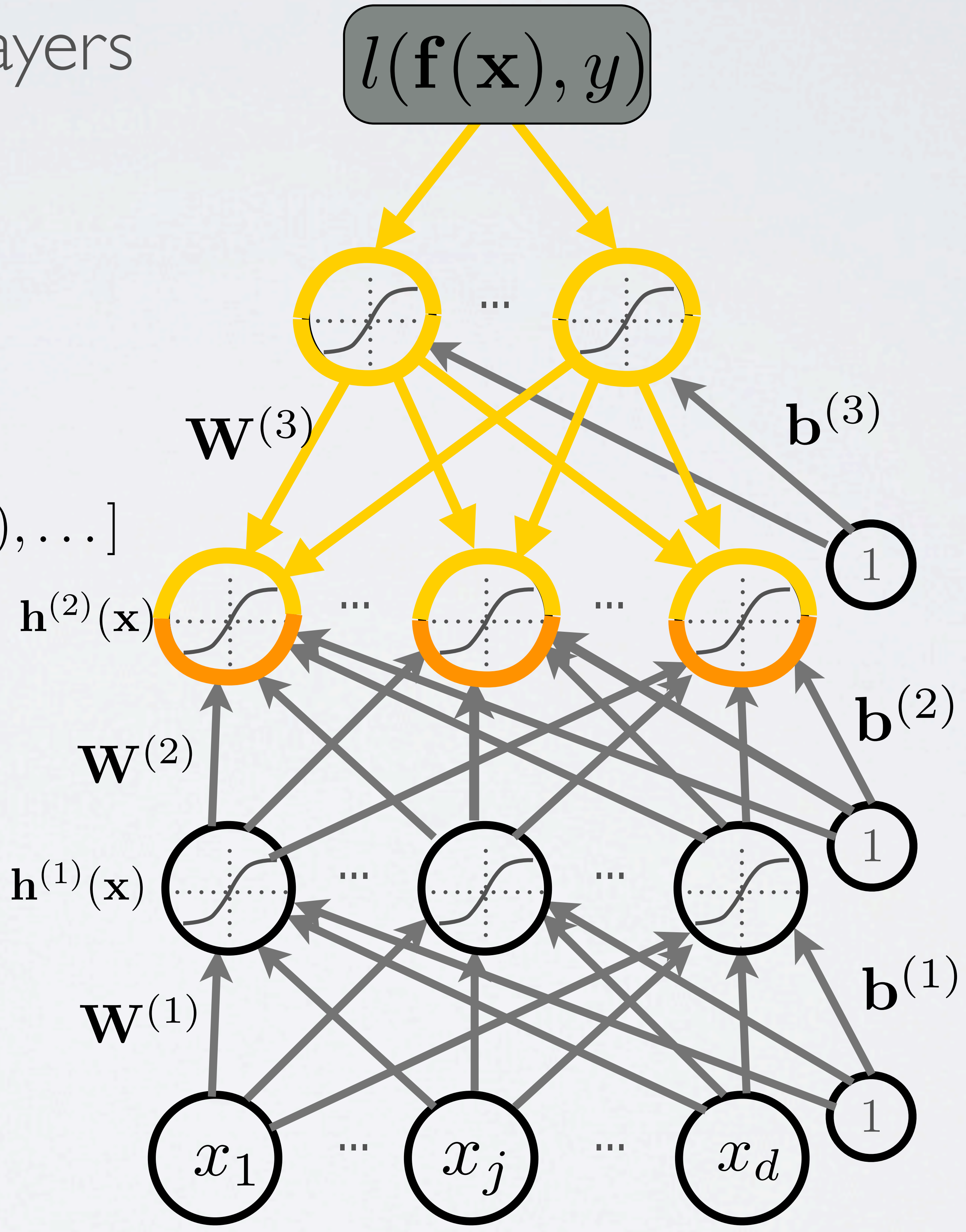
GRADIENT COMPUTATION

Topics: loss gradient at hidden layers
pre-activation

- Gradient:

$$\nabla_{\mathbf{a}^{(k)}(\mathbf{x})} - \log f(\mathbf{x})_y$$
$$= \left(\nabla_{\mathbf{h}^{(k)}(\mathbf{x})} - \log f(\mathbf{x})_y\right)^\top \nabla_{\mathbf{a}^{(k)}(\mathbf{x})} \mathbf{h}^{(k)}(\mathbf{x})$$
$$= \left(\nabla_{\mathbf{h}^{(k)}(\mathbf{x})} - \log f(\mathbf{x})_y\right) \odot [\ldots, g'(a^{(k)}(\mathbf{x})_j), \ldots]$$

element-wise product



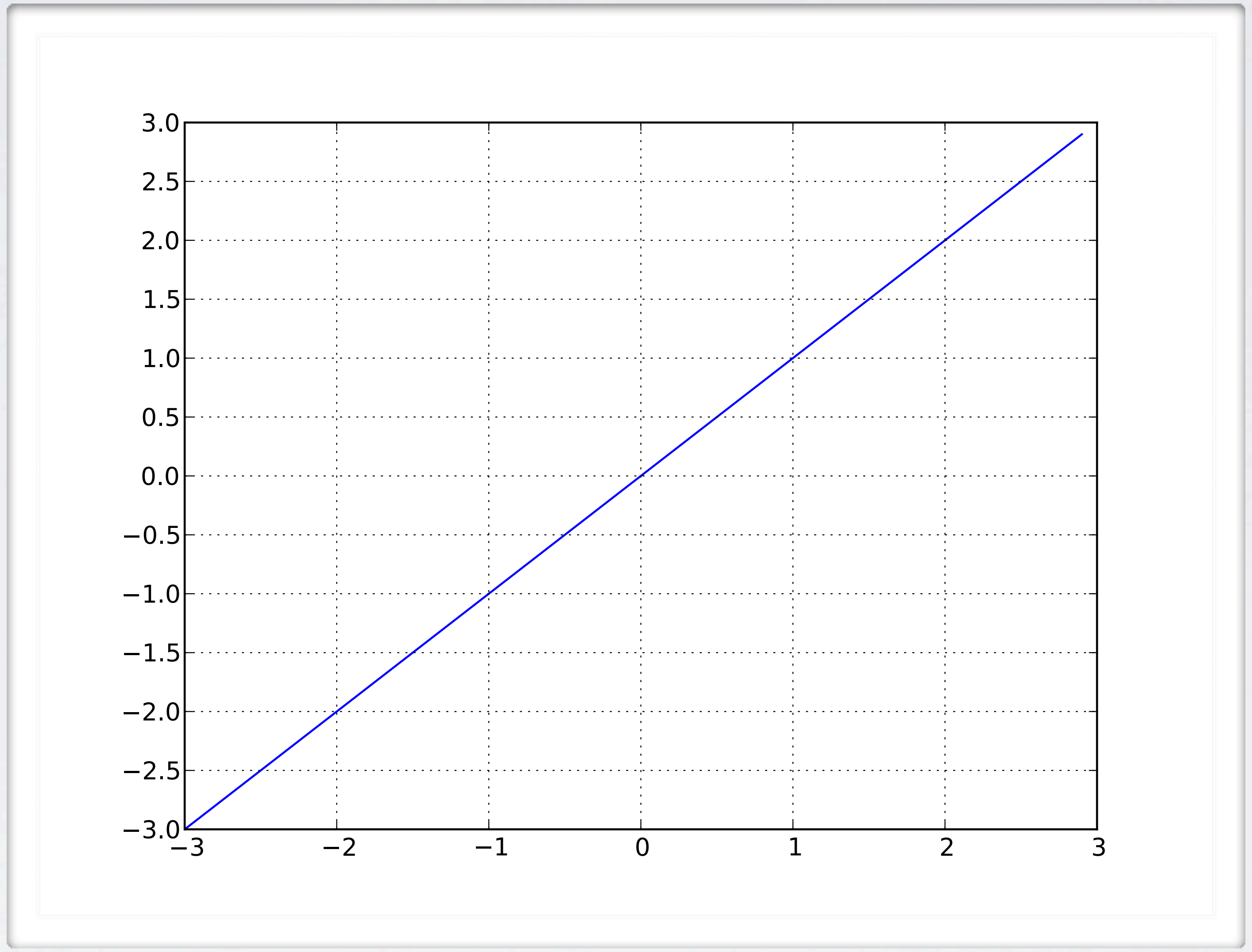
REMINDER
$$h^{(k)}(\mathbf{x})_j = g(a^{(k)}(\mathbf{x})_j)$$

ACTIVATION FUNCTION

Topics: linear activation function gradient

- Partial derivative:

$$g'(a) = 1$$



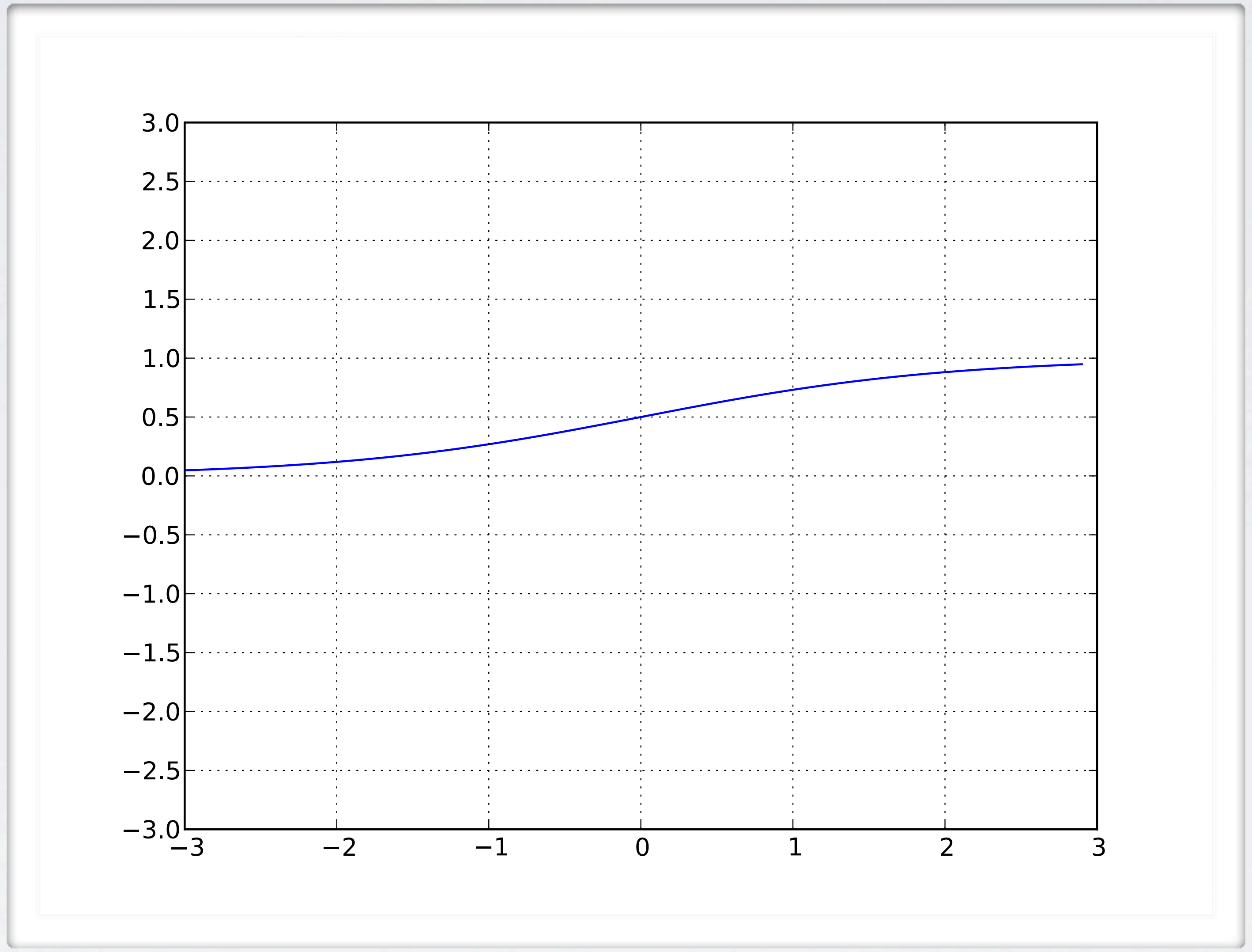
$$g(a) = a$$

ACTIVATION FUNCTION

Topics: sigmoid activation function gradient

- Partial derivative:

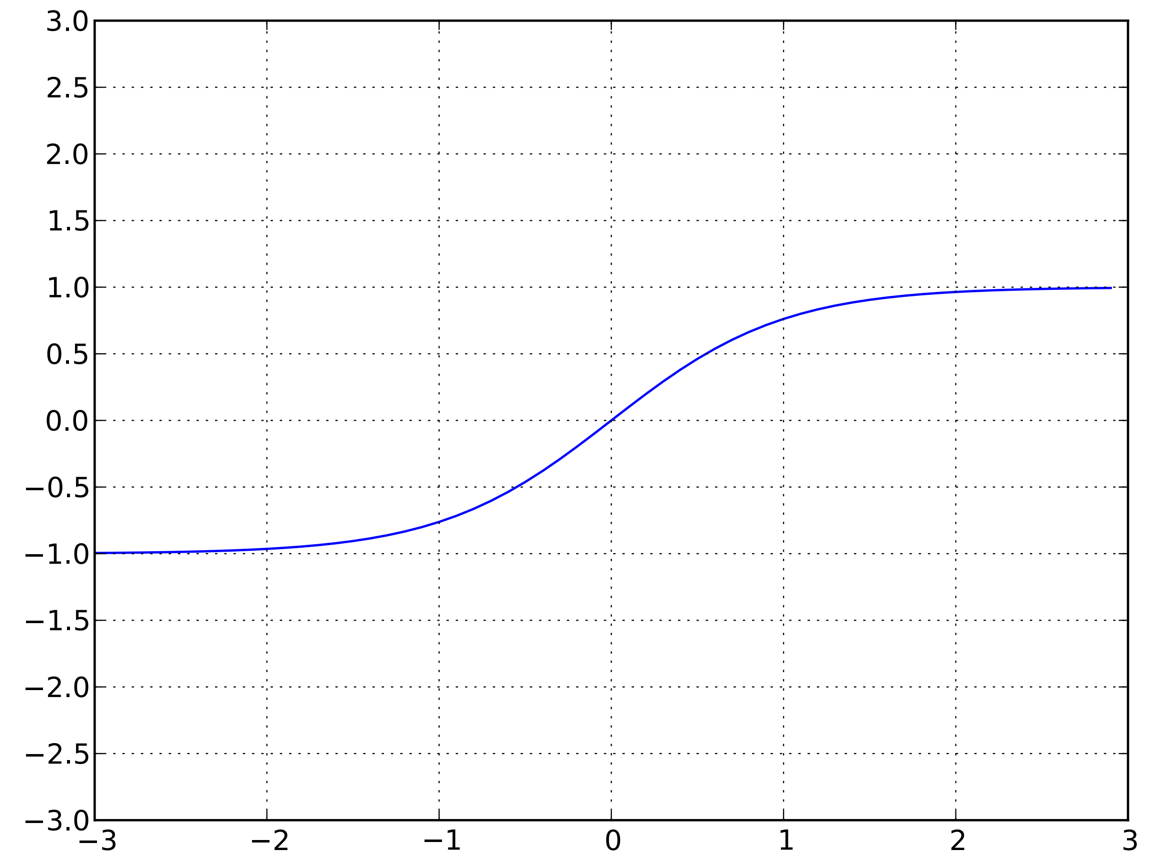
$$g'(a) = g(a)(1 - g(a))$$



$$g(a) = \mathrm{sigm}(a) = \frac{1}{1+\exp(-a)}$$

ACTIVATION FUNCTION

Topics: tanh activation function gradient



- Partial derivative:

$$g'(a) = 1 - g(a)^2$$

$$g(a) = \tanh(a) = \frac{\exp(a)-\exp(-a)}{\exp(a)+\exp(-a)} = \frac{\exp(2a)-1}{\exp(2a)+1}$$